Learning with Missing Data

O. Ozan Koyluoglu and Naveen Ramakrishnan

Abstract

This work considers the problem of learning with missing data. Two main classes of approaches are considered. The first class consists of sequential algorithms in which the missing values are first imputed by using an imputation method and then a learning algorithm is applied. This sequential approach is shown to be non-robust for certain scenarios. The second class of algorithms is more robust as they allow exploitation of side information (location of missing values) from the imputation block, which enhances the performance. In particular, an online updation scheme is proposed which is computationally efficient.

I. Introduction

The problem of missing data in observations is a common phenomenon in statistics [1], [2]. For statistical learning applications, missing data pose a greater challenge since the performance of any regression/classification algorithm will be affected. In fact, not only testing data but also training data might contain missing values in many applications.

Some of the motivations for the missing training data in learning are as follows: In face recognition applications, partial occlusions pose a great problem for the underlying algorithms as we do not have values for most of the features [3]. Another commonly encountered case is the processing of survey datasets in which some questions might be optional or intentionally left blank [4]. Even some datasets might contain unreliable (or noisy) measurements. For example, in sensing applications there might be faulty sensors or burst errors which need to be taken care of. Learning with missing data also has extensive applications in financial and stock market datasets [5], [6]. Another interesting and crucial case is the following: Lets imagine a scenario where data is collected from companies which consider different set of features to be sensitive. In this scenario, those features may not be available for the learning algorithm which motivates us to consider the missing data problem for learning algorithms. Missing data problem is mainly studied in bio-statistics where microarray gene expression data contain several missing values [7]–[9]. In all these works, the authors consider only imputation of the missing values and do not analyze the effect of imputation on the performance of the learning algorithms. However this constitutes the main theme of our work.

We now give some notational remarks that we use throughout the rest of the sequel and comment on how we can model the missing data. Consider a data set with n observations, p features, and a response as depicted in Fig. 1. The data matrix is represented with the matrix $\mathbf{X}$, in which the value of x_{ij} might be missing. There are two

The authors are with the Department of Electrical and Computer Engineering Ohio State University, Columbus, Ohio. Email: {koyluogo, ramakrin}@ece.osu.edu. This work is submitted on March 16, 2009 as partial fulfillment of the requirements for the STAT 760, Winter 2009.

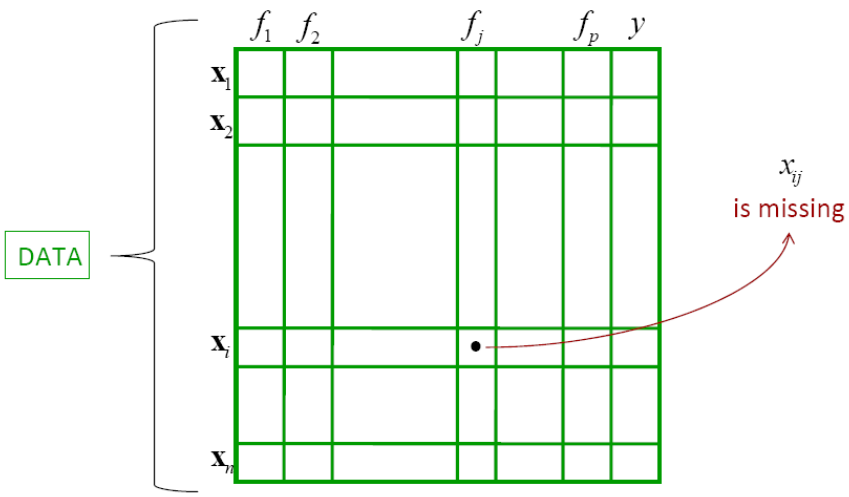

Fig. 1. Missing data.

main techniques in modeling missing values [1]. Let $\mathbf{M}$ be the missing data indicator matix, i.e. $m_{ij} = 1$, if x_{ij} is missing. Let $\mathbf{X}_c$ denote the complete dataset and ϕ denote the unknown parameter (according to some model) characterizing missingness of the data. The first model can be represented by

$$P(\mathbf{M}|\mathbf{X}_c, \phi) = P(\mathbf{M}|\phi). \tag{1}$$

That is the missing value locations are independent of the complete data set given the missingness parameter ϕ. This model is called missing completely at random (MCAR). For the second model, we first denote the missing value matrix by $\mathbf{X}_m$ (this can be considered as the difference between $\mathbf{X}_c$ and $\mathbf{X}$). Then, the second model assumes

$$P(\mathbf{M}|\mathbf{X}_c, \phi) = P(\mathbf{M}|\mathbf{X}, \phi), \quad \forall \mathbf{X}_m, \phi. \tag{2}$$

This model is called missing at random (MAR), where the missing value locations are independent of the missing values once the remaining values ($\mathbf{X}$) and the missingness parameter (ϕ) are given. In this sequel, we focus on MCAR, where the missing value locations are totally independent from the dataset.

The rest of this report is organized as follows. Section II introduces sequential algorithms for learning with missing data, where we first consider simple imputation methods and then use one of the known learning algorithms. This section is concluded with numerical results, where we compare the performance of various sequential algorithms. Section III focuses on side information based learning algorithms, where we propose algorithms that exploit the location of missing values in learning. The simulation results in that section shows that the proposed schemes are robust and performs well even for the datasets having antipodal important features. Finally, we provide some concluding remarks in Section IV.

II. Sequential Approaches to Learning with Missing Data

In this section, we focus on sequential approaches to learning with missing data. In such approaches, the missing values of the data is first imputed using the available data. And then, some learning algorithm is applied. See Fig. 2, where a block diagram of the sequential approach is shown. The approach is sequential in the sense that the learning block is independent of the imputation block.

Fig. 2. Sequential approach.

A. Algorithms

A naive approach to the missing data problem is to omit the samples which have missing values and treat the rest as the training set. See Fig. 3, where we provide a toy example with $p = 3$ and $n = 5$. The limitation of this

Fig. 3. Deletion of observations. Rows two and four have missing values and hence are deleted.

approach is that it throws away valuable information required for learning.

Another similar method is deletion of features with missing values. See Fig. 4. The limitation of this approach might be more severe as the data dimension is reduced. On the contrary, if the deleted feature has a weak correlation with the response variable, the loss of that feature does not affect the performance much. This method can also be related to subset selection methods. For example, we may consider the features without missing values first and then do forward selection to add new features after estimating their missing values. See Fig. 4.

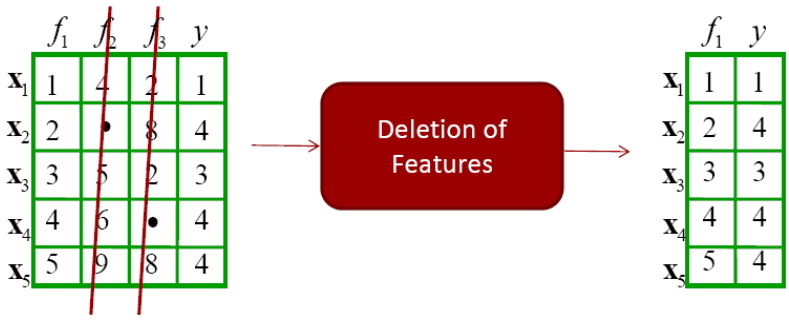

Fig. 4. Deletion of feature. Columns two and three have missing values and hence are deleted.

The third approach is to average the observed data in the corresponding feature to fill the missing values. Focusing on column j and denoting missing rows by $\mathbf{i}_m$, the algorithm fills the missing values according to the following.

$$\mathbf{X}(\mathbf{i}_m, j) = \frac{1}{n - |\mathbf{i}_m|} \sum_{i \notin \mathbf{i}_m} \mathbf{X}(i, j) \tag{3}$$

See Fig. 5. One can claim this to be optimal if the features are independent and fraction of missing data is less compared to the number of training samples. For classification problems, this naive scheme can be improved by restricting the average among the class labels. This method is also called mean imputation in [10]. One general

Fig. 5. Averaging.

algorithm is the Expectation Maximization which is used to fill the missing values [11]. This method works by maximization of the expected log-likelihoods in each step.

The mean imputation method can also be modified by only averaging over K-nearest neighbors (K-NN), where the distance between samples can be computed with some distance metric (Euclidean norm is used throughout the sequel) and neighbors not having missing data could be considered. Here, for observation i, focusing on the missing value column, say column j, and denoting rows of K nearest neighbors by $\mathbf{i}_{NN}$, the algorithm fills the missing

values according to the following.

$$\mathbf{X}(\mathbf{i}_{NN}, j) = \frac{1}{n - |\mathbf{i}_{NN}|} \sum_{i \in \mathbf{i}_{NN}} \mathbf{X}(i, j) \tag{4}$$

See Fig. 6 for a toy example. The main advantage of K-NN over the mean imputation method is that K-NN uses

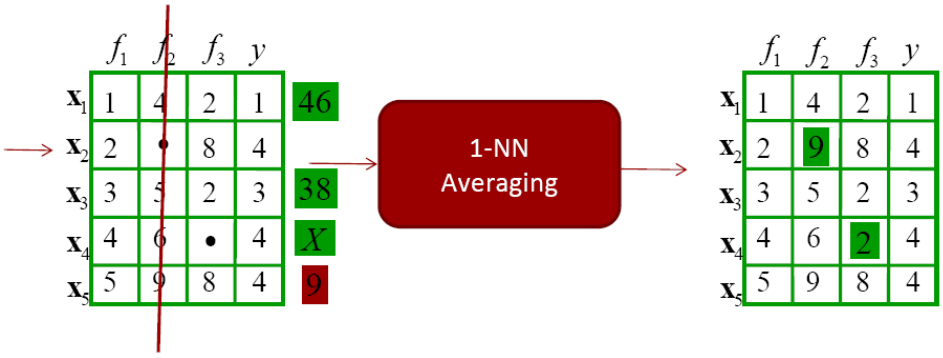

Fig. 6. K-nearest neighbor averaging with $K = 1$. To impute $\mathbf{X}(2, 2)$, the nearest neighbor is found to be row 5, which results in 9 to be the imputed value. Similarly, $\mathbf{X}(4, 3)$ the value of 2 is imputed.

the correlation between the features to impute the missing values, whereas the mean imputation relies only on the corresponding feature.

After applying one of these imputation methods, the data could be considered as full and then known techniques for the learning problem can be applied. In this report, we focus on three regression methods in the context of missing data: Least Squares, Ridge Regression, and Lasso. These methods are applied after using one of the three imputation methods, namely 1) Deletion of observations, 2) Deletion of features, 3) Averaging, and 4) K-nearest neighbor averaging. These sequential learning approaches, i.e., first imputing the data and then running learning algorithms, lead us to analyze 12 distinct algorithms, each using an imputation method and a learning method. In the the next section, we analyze the performance of theses schemes with numerical experiments, where we also compare the results obtained by using the full data.

B. Numerical Results

To obtain numerical results for the missing data problem, we first introduce two types of deletion methods. The first one is the deletions based on observation for which we define the deletion vector as below.

$$f_o = [f_{o1}, f_{o2}, \ldots, f_{oL}], \quad \sum_l f_{ol} \leq 1, \tag{5}$$

where f_{ol} is the frequency of observations with l missing values and $L \leq p$. For example, $f_o = [0.20.1]$ means 20% of the randomly-selected observations have 1 missing value in any of the features and 10% of the randomly-selected observations have 2 missing values. While using the deletion vector, we omit the rest of the frequencies if they are zero.

The second type is the deletions based on features in which we have the following deletion vector,

$$f_f = [f_{f1}, f_{f2}, \ldots, f_{fp}], \tag{6}$$

where f_{fj} denotes the missing frequency for feature j.

These deletion methods are then used to obtain the data with missing values, which are used for studying the performance of the aforementioned algorithms.

We use the Boston housing data, which has $p = 13$ features and $N = 506$ observations. We randomly reserve 106 of them for testing and used remaining 400 of them for training. We introduce missing values in the data using one of the aforementioned deletion methods. After obtaining the data with missing values, we simulated our imputation methods and then run the learning algorithms (LS, ridge regression, and lasso). We implemented V-fold cross validation for the ridge regression and lasso algorithms.

The deletion of observations, deletion of features and averaging are easy to implement whereas the K-nearest neighbor needs to be carefully simulated. For implementing the K-nearest neighbor algorithm, we pick each observation and then determine whether it has missing data or not. If the observation chosen does not have any missing value, we move to the next observation otherwise we find the positions of the missing values. Then we delete all the columns in the remaining data corresponding to the positions of the missing data in the chosen observation and also remove any row with missing values in the remaining columns. Next we determine the distance between the chosen observation and each of the remaining observations using the remaining columns. For each missing value in the chosen observation we average the values in the corresponding column of the K-nearest neighbors by ensuring that the neighbors do not have any missing value in the corresponding columns. We repeat this for each missing value in the chosen observation and then move on to the next observation.

We then run the above experiment many times and average the resulting mean square error (MSE) values. For comparison, we also implemented the above learning algorithms with the full data.

Our first experiment is with the Boston Housing data and using deletions based on observations. The simulation results are shown in Fig. 7, where we choose $K = 4$ for the K-NN. For the Boston Housing dataset and the nature of random deletions, full data performs better than the other algorithms. It is also observed that averaging is closest to full data for the chosen deletion vectors and deletion of observation performs the worst since we throw important information. Here we cannot implement deletion of features since we will end up deleting all the features if each feature contains at least one missing value. For a given imputation method learning algorithms (LS, Ridge and Lasso) have similar performance. We repeat the same experiment by introducing many missing values. The results are shown in Fig. 8. The deletion of observation method only depends on the number of deleted observations as can be seen from the Figures 7 and 8 and does not vary as we increase the number of missing values in a given observation as expected. For this dataset and the choice of deletion vectors, K-NN is worse than averaging. In this experiment, the number of missing values is much more compared to the previous one and hence the performance difference between averaging and full data is significant. However, we note that these deletion vectors are not practical as they contain too many missing values per observation, which explains the poor

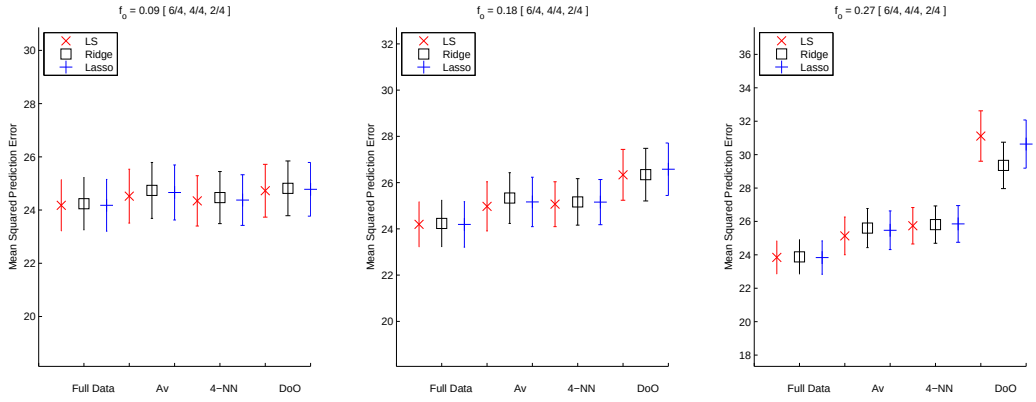

Fig. 7. Numerical results for the three different deletion (based on observation) vectors.

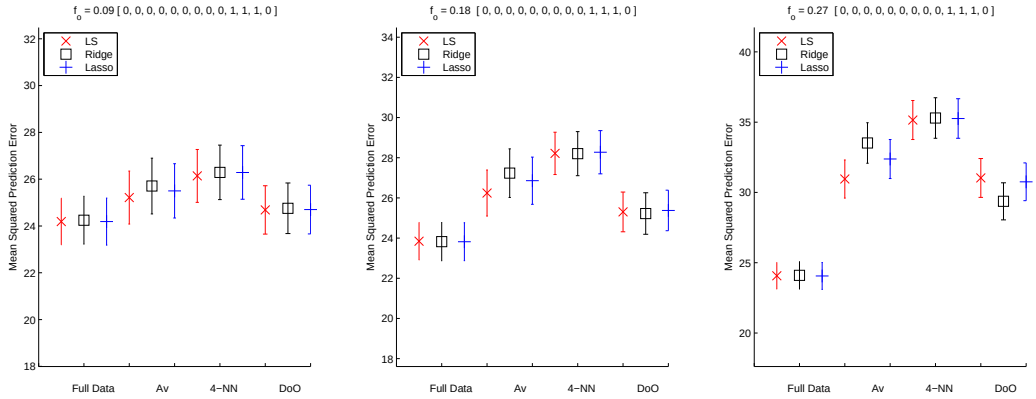

Fig. 8. Numerical results for the three different deletion (based on observation) vectors.

performance of averaging compared to deletion of observation.

The next experiment is for deletion based on feature, for two different deletion vectors. The results are shown in Fig. 9 from which similar observations can be made. For this experiment, we do not implement the deletion of observation method since we might end up deleting all observations. To probe further, we use a different deletion vector in Fig. 11 where the deletions are introduced for the most important feature f_{13}. For Boston Housing dataset the importance of each feature can be analyzed by ridge coefficients for the full dataset which is plotted in Fig. 10. As we can note from Fig. 11, the performance of the deletion of feature degrades much more compared to other algorithms for this choice of deletion vector as the most important feature is deleted. This experiment also shows that K-NN can perform better than averaging in certain scenarios.

To study the effect of the value K in K-NN averaging, we simulated Boston Housing data with deletions based on observation by varying K. Fig. 12 shows the results in which we can see that K-NN meets the performance of

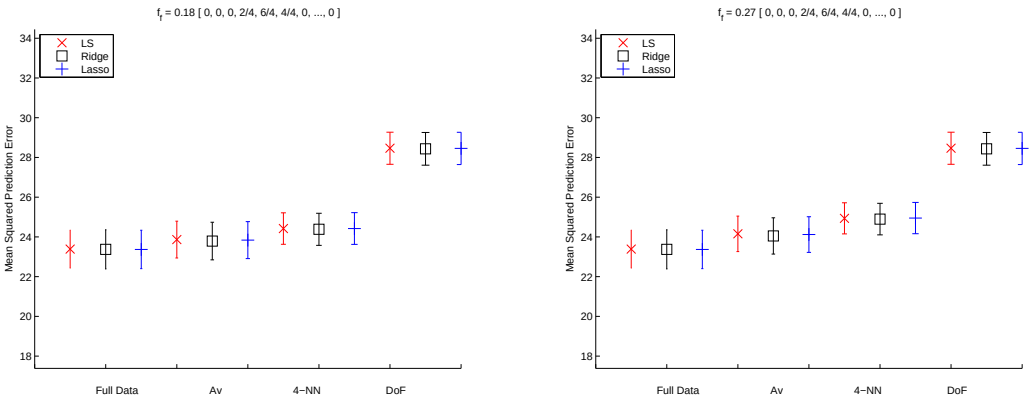

Fig. 9. Numerical results for the two different deletion (based on feature) vectors.

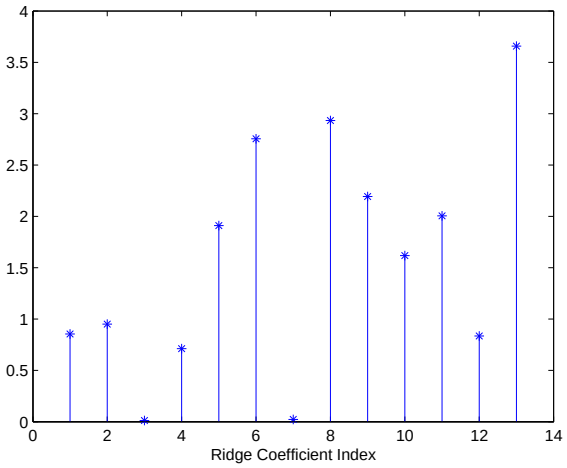

Fig. 10. Absolute values of the ridge coefficients for the full dataset (Boston Housing).

averaging as K increases.

Till now, the performance of averaging was the best (except the unpractical deletion vector case). But this need not be true especially in cases where the important features have antipodal-like densities. Here we analyze synthetic data where we introduce antipodality and correlations between the features. In particular, we choose $p = 13$ and $n = 506$ and fill in the values of the data matrix with i.i.d. $\sim \mathcal{N}(0, 1)$ entries. Then, we introduce correlation and antipodality, and choose the response variable as follows.

$$\begin{aligned} x_1 &= 2\text{sign}(x_4), \quad x_3 = 2\text{sign}(x_6) \\ y &= \sum_{j=1}^{p} \frac{10}{j^2} x_j + x_1^2 + 2x_2^2 \end{aligned} \tag{7}$$

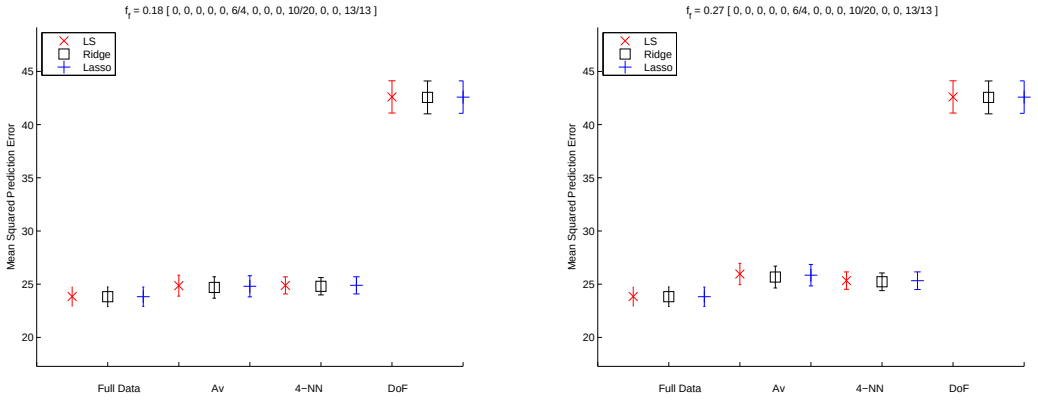

Fig. 11. Numerical results for the two different deletion (based on feature) vectors.

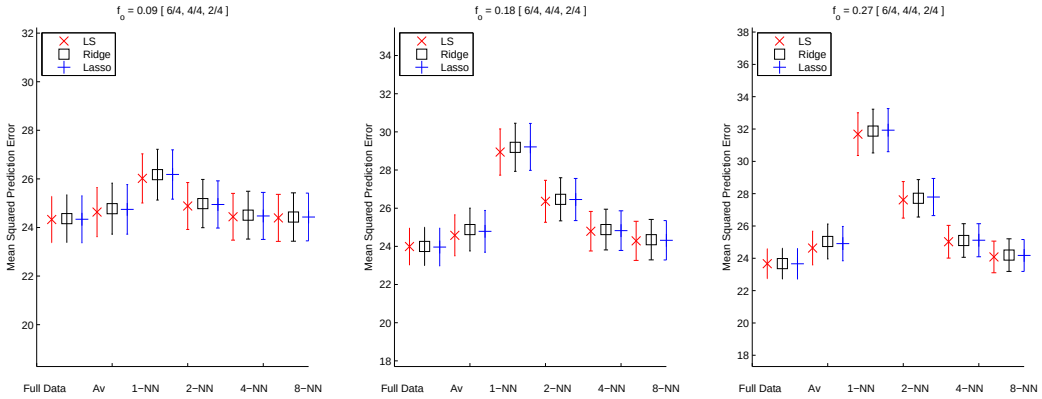

Fig. 12. Effect of K in K-NN averaging for Boston Housing dataset.

We introduce non-linearity in the response so that the MSE will not become zero. Fig. 13 shows the results for deletions based on features. Here we notice that averaging performs the worst and and intermediate value ($K = 4$) is the optimum choice for K-NN.

The above results indicate that the sequential approaches considered are not robust (according to datasets). Hence new techniques are needed for ensuring robustness, which is the topic of discussion in the next section.

III. Side-Information Based Learning with Missing Data

In the previous section learning and imputation were independent of each other. Here we note that the learning algorithms can exploit the information about the locations of the missing values. A block diagram of this approach is depicted in Fig. 14.

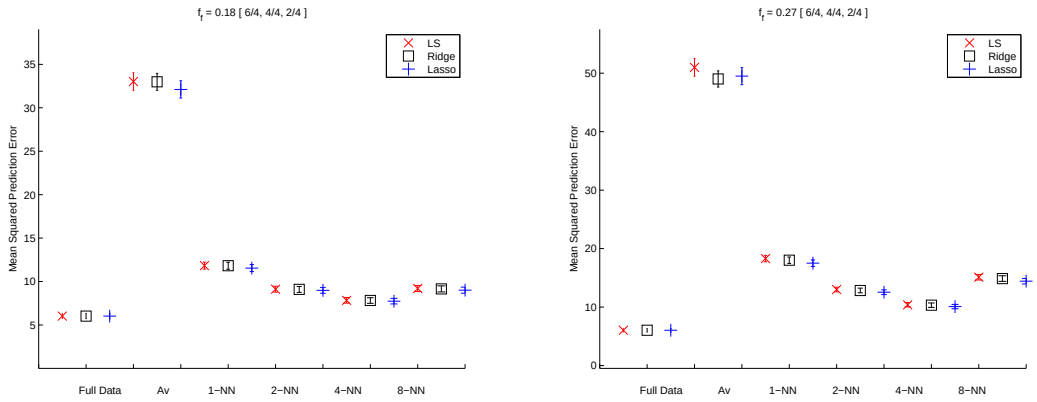

Fig. 13. Effect of K in K-NN averaging for Synthetic data.

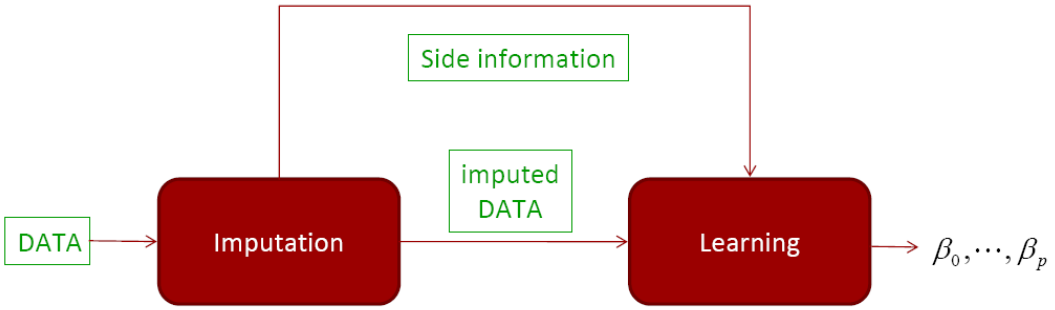

Fig. 14. Side-information based learning.

A. Proposed Algorithms

In this subsection we propose three side-information based learning algorithms which exploit the missing value locations. For all the three algorithms, instead of naive imputation methods described in the previous section, we used learning algorithms for imputation.

In this imputation method, we first order the observations according to the number of missing values. Consider observation $\mathbf{x}_i$ having a missing value at feature j. We then fit a linear model (using ridge) by considering f_j as the response variable and all the other columns (including y but excluding columns where $\mathbf{x}_i$ has a missing value) till $\mathbf{x}_{i-1}$ as the predictors. We use the fitted model to fill the missing values (See Fig. 15). We repeat the procedure for every other missing value for that observation. A similar imputation method is independently considered in [12], where the authors use repeated regressions (regression method has not been specified) for imputation of missing data in gene expression arrays.

We now describe how side information can be used in learning algorithms. In our first approach (PA1), we

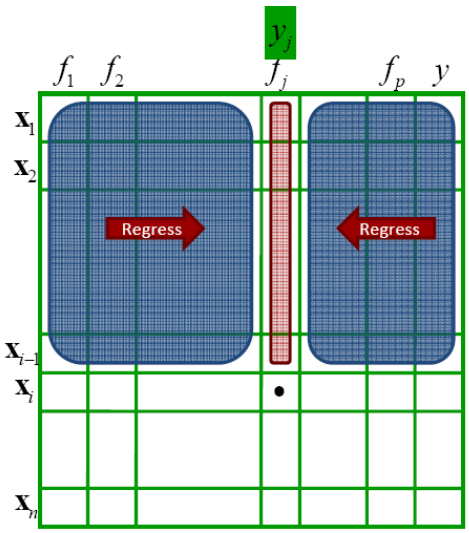

Fig. 15. Regression based imputation.

associate a cost to each observation by counting its number of missing values, say c_i. Then we use a function, $g_1(c_i)$, to scale down each observation ($\mathbf{x}_i$). The modified dataset is used in ridge regression for which the residual sum of squares (RSS) can be represented as follows.

$$\text{RSS}_{\lambda,\mathbf{c}}(\boldsymbol{\beta}) = \sum_{i=1}^{n} g_1(c_i)\Big(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij}\Big)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \tag{8}$$

Here the function $g_1(c_i)$ should be chosen as a decreasing function of c_i. For all the simulation in this section, we consider $g_1(c_i) = \frac{1}{2c_i}$. We set $g_1(c_i) = 1$ for observations without missing values.

Our second approach (PA2) also has the same pre-processing of data for imputation. Next we associate a cost to each feature by counting the number of missing values, say c_j for that feature. We then use the following modified form of ridge regression using these costs.

$$\text{RSS}_{\lambda,\mathbf{c}}(\boldsymbol{\beta}) = \sum_{i=1}^{n} \Big(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij}\Big)^2 + \lambda \sum_{j=1}^{p} g_2(c_j)\beta_j^2 \tag{9}$$

The cost function $g_2(c_j)$ is an increasing function of c_j which allows us to penalize unreliable features more than the reliable ones. We use $g_2(c_j) = c_j$ for the simulations. Compared to PA1, here we penalize each feature instead of each observation.

The last approach (PA3) is an online (case-by-case) learning algorithm which can reduce the complexity compared to the previous approaches. Unlike PA1 and PA2, this approach does not require a separate learning step after imputation. In PA3, we first fit a linear model (using ridge) with observation without missing values, i.e. we compute $\boldsymbol{\beta}$. Then for observation $i+1$, we first impute the missing values (using ridge), compute its associated

cost $g_3(c_{i+1})$, and then update the coefficients with the following equation.

$$\boldsymbol{\beta}^{i+1} = (1 - g_3(c_{i+1}))\boldsymbol{\beta}^i + g_3(c_{i+1})\hat{\boldsymbol{\beta}}^{i+1}, \tag{10}$$

where

$$\boldsymbol{\beta}^{i+1} = \arg\min_{\boldsymbol{\beta}} \sum_{k=1}^{i+1} \left(y_k - \beta_0 - \sum_{j=1}^{p} \beta_j x_{kj}\right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \tag{11}$$

We notice that we need to solve the optimization problem in (11) for each new observation $i+1$. But this can be avoided by efficiently updating the $\hat{\boldsymbol{\beta}}^{i+1}$ using $\hat{\boldsymbol{\beta}}^{i}$ and $\mathbf{x}_{i+1}$ by exploiting the matrix inversion lemma [13].

$$\begin{aligned} \hat{\boldsymbol{\beta}}^{i+1} &= \mathbf{E}_{i+1}\hat{\boldsymbol{\beta}}^{i} + y_{i+1}\mathbf{E}_{i+1}\mathbf{A}_i^{-1}\mathbf{x}_{i+1}^T \\ \mathbf{A}_{i+1}^{-1} &= \mathbf{E}_{i+1}\mathbf{A}_i^{-1} \\ \mathbf{E}_{i+1} &= \mathbf{I} - \frac{1}{1 + \mathbf{x}_{i+1}\mathbf{A}_i^{-1}\mathbf{x}_{i+1}^T}\mathbf{A}_i^{-1}\mathbf{x}_{i+1}^T\mathbf{x}_{i+1}\mathbf{A}_i^{-1} \end{aligned} \tag{12}$$

For all the simulation in this section, we consider $g_3(c_i) = \frac{1}{2c_i}$. We set $g_3(c_i) = 1$ for observations without missing values.

B. Numerical Results

In this section, we compare the sequential approaches mentioned in the previous section (we used ridge regression as the underlying learning learning algorithm) with the proposed approaches above. First experiment results are given in Fig. 16. For this set of deletion vectors, we observe that PA2 and PA3 perform as good as averaging and K-NN, which are close to the MSPE of the full dataset. For higher deletion frequencies, PA2 and PA3 can even perform better than averaging and K-NN.

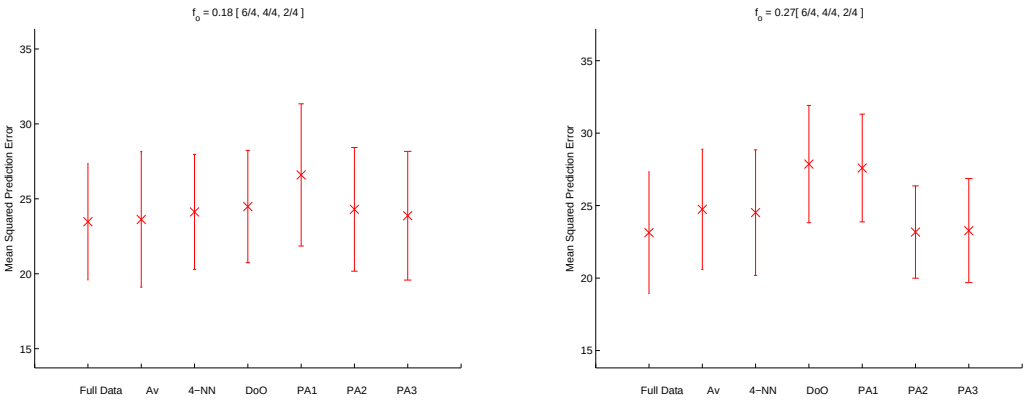

Fig. 16. Numerical results for new algorithms using Boston Housing dataset and deletions are based on observations.

We then repeat the experiment with the synthetic data described in the previous section. The results are shown in Fig. 17. We see that PA2 and PA3, unlike averaging, are robust to antipodality inherent in the important features.

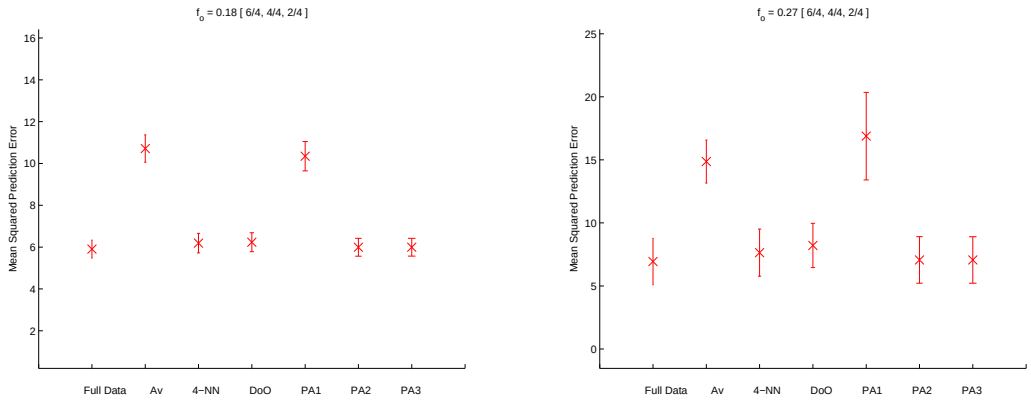

Fig. 17. Numerical results for new algorithms using Synthetic data and deletions are based on observations.

In Figs. 18-20, we provide numerical results for the deletions based on features. In Fig. 18, we observe that PA2 and PA3 perform similar to averaging and K-NN. PA1 performs similar to deletion of features. In Fig. 19, now PA1 performs a lot better compared to deletion of features. This is expected, as the deletion of the most important feature degrades the performance, whereas PA1 does not discard the feature but instead penalizes it. We consider the synthetic data in Fig. 20, where averaging does not perform well. Here, MSPE of K-NN and PA1 are slightly above that of full data, whereas that of PA2 and PA3 perform as if there are no missing values. This confirms the robustness of the proposed algorithms PA2 and PA3. We note that, one can further think of different cost functions for the proposed algorithms. One can even choose one of the cost functions by using model selection tools. Finally, we remark that by doing so one can get performance enhancements for the proposed algorithms.

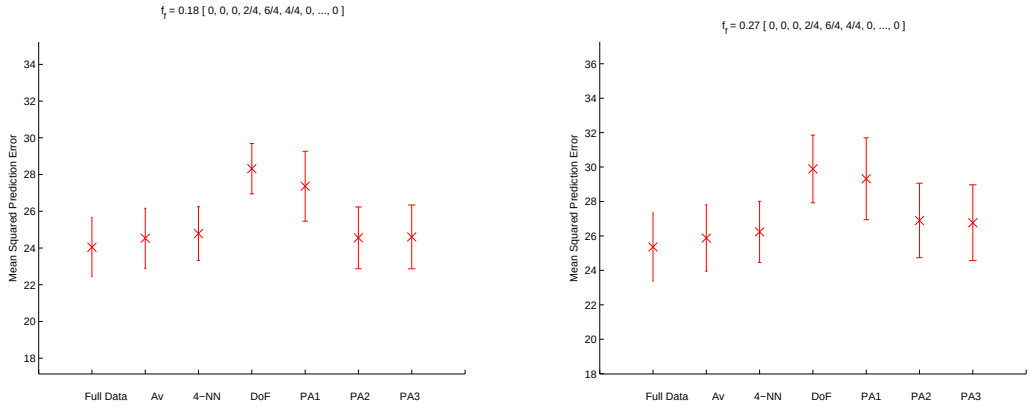

Fig. 18. Numerical results for new algorithms using Boston Housing dataset and deletions are based on features.

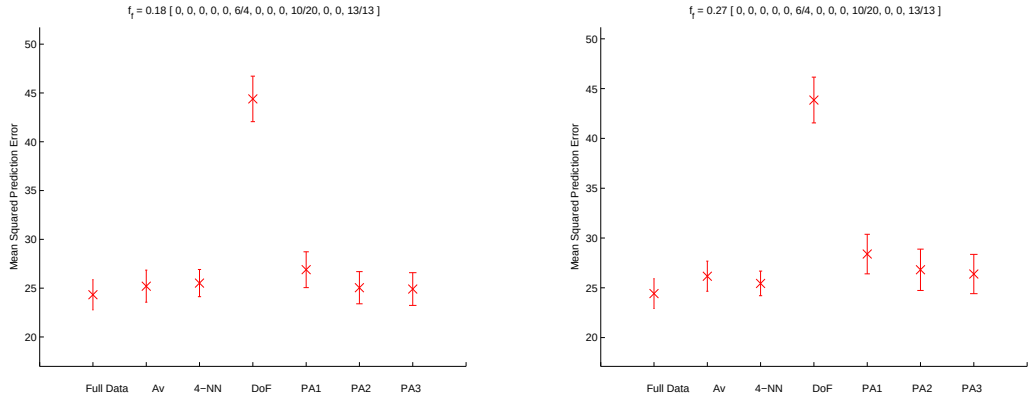

Fig. 19. Numerical results for new algorithms using Boston Housing dataset and deletions are based on features.

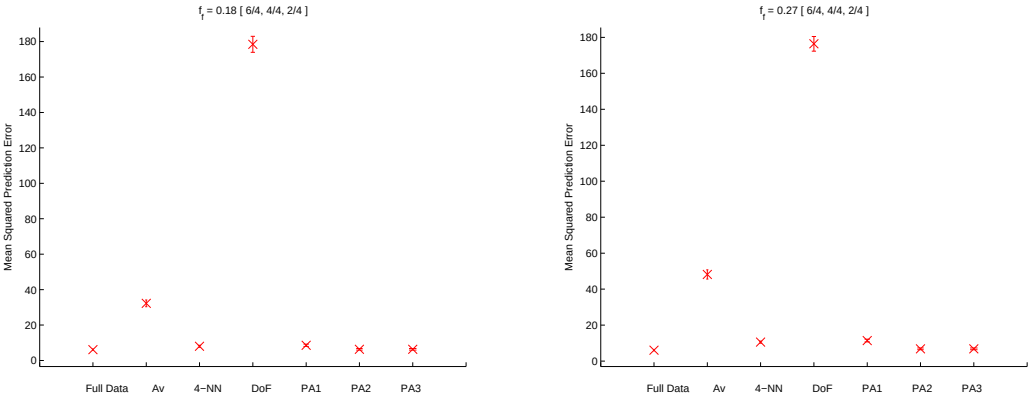

Fig. 20. Numerical results for new algorithms using Synthetic data and deletions are based on features.

IV. Concluding Remarks

In this work, we have focused on the problem of learning with missing data. Our focus mainly consists of two approaches to the problem. Firstly, we analyzed sequential approaches, where the algorithms first use some imputation technique to fill the missing values and then apply a learning algorithm to find out an underlying linear model. For the imputation task, we consider approaches like deletion of observations, deletion of features, averaging, and K-nearest neighbor averaging; and the learning algorithms include least squares (LS), ridge regression, and lasso. Several numerical experiments using Boston Housing and Synthetic datasets are conducted, which show us that the sequential algorithms are not robust against antipodal important features. To overcome this limitation, we next propose learning algorithms that exploit the missing value locations in the dataset. Basically, the sequential approach we consider involved two independent blocks (imputation and learning), whereas the proposed techniques exploit side information (location of the missing values) from the imputation method in the learning process. The

main advantage of the proposed schemes are two-fold. First, they use regression techniques (ridge) to impute the data compared to approaches like averaging. Second, the algorithms associate cost functions for observations/features which are then used in the learning algorithms. This allows us to penalize unreliable observations/features in the learning process. Finally, the last proposed approach involves an online update rule which can update the coefficients of the model by using one observation at a time. This can be used in decreasing the complexity of the underlying learning task in situations where observations are obtained in real time.

We note possible future research directions here. Remarkable, one can readily extend the proposed algorithms to multi-response cases. Also, the last algorithm can be modified to have update rule with features, in which we add features one by one to the model after estimating the missing values. Studying other datasets to compare the performance of the algorithms is of definite interest. In particular, datasets involving antipodal important features and other correlation structures can be considered. In addition, we note that regression problem is considered in this work. Missing data in classification problems is also an important one to study. Also, we remark that we do not consider missing values in test observations and this may be an interesting direction for future study. Finally, one can study the interplay between missing value problem and cost based learning approaches. In such a scenario, we might have cost of features/observations according to their reliability and our task may include to choose best set of features/observations that lead to the best model for the data.

REFERENCES

[1] R. J. A. Little and D. B. Rubin, ”Statistical Analysis with Missing Data,” NY: Wiley, 1987.

[2] D. Rubin, ”Inference and missing data (with discussion),” *Biometrika*, vol. 63, pp. 581-592, 1976.

[3] H. Jia and A. Martinez, ”Face Recognition with Occlusions in the Training and Testing Sets,” in *Proc. of the 8th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2008)*, Amsterdam, The Netherlands, 2008.

[4] A. Alfons, ”Complex Survey Data Sets: Visualization of Missing Values in R,” *Young European statisticians Workshop (YES-II), ”High dimensional statistics”, EURANDOM*, Eindhoven, The Netherlands, October 6-8, 2008.

[5] R. A. Jarrow, ”Finance Theory,” Prentice-Hall, Inc., 1988.

[6] W. F. Sharpe, G. J. Alexander, and J. V. Bailey, ”Investments,” 6th ed., Prentice-Hall, Inc., 1999.

[7] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. B. Altman, ”Missing value estimation methods for DNA microarrays,” Bioinformatics vol. 17, no. 6, pp. 520525, 2001.

[8] Hyunsoo Kim, Gene H. Golub, and Haesun Park, ”Missing value estimation for DNA microarray gene expression data: local least squares imputation,” Bioinformatics, vol. 21, no. 2, pp. 187-198, Jan. 2005.

[9] Xiaobai Zhang, Xiaofeng Song, Huinan Wang, and Huanping Zhang, ”Sequential local least squares imputation estimating missing value of microarray data,” Computers in Biology and Medicine, vol. 38 , no. 10, Oct. 2008.

[10] T. Hastie, R. Tibshirani and J. Friedman, ”The Elements of Statistical Learning: Data Mining, Inference, and Prediction,” 2nd ed., Springer Series in Statistics, 2009.

[11] A. P. Dempster, N. M. Laird, and D. B. Rubin, ”Maximum Likelihood from Incomplete Data via the EM Algorithm,” *Journal of the Royal Statistical Society*, Series B, Vol. 39, No. 1, pp. 1-37, 1977.

[12] T. Hastie, R. Tibshirani, G. Sherlock, M. Eisen, P. Brown, and D. Botstein, ”Imputing Missing Data for Gene Expression Arrays,” Technical Report, Division of Biostatistics, Stanford University, 1999.

[13] R. A. Horn and C. R. Johnson, ”Matrix Analysis,” Cambridge University Press, 1985.